2

High Performance Video Computer

S. P. Knight
Group Head
Parallel Computing Research

David Sarnoff Research Center
CN-5300
Princeton, NJ 08543-5300

Contract No. MDA972-90-C-0022

Quarterly Technical Report No. 1
Report Period: February 1, 1990 to October 15, 1990

November 5, 1990

Prepared for
DARPA
1400 Wilson Blvd.
Arlington, VA 22209-2308

Notice - Restricted Rights

Use, duplication, or disclosure is subject to restrictions stated in Contract No. MDA 972-90-C-0022 with the David Sarnoff Research Center, Inc.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either written or implied, of the Defense Advanced Research Projects Agency or the U. S. Government.

Table of Contents

Section **Page**

A. Task Objectives 1

B. Technical Problem 1

C. General Methodology 1

D. Technical Results 2

1. 1024 Processor Princeton Engine for NIST 3
2. Fortran Compiler for the Princeton Engine 3
3. Sarnoff Engine Design 4

E. Important Findings and Conclusions 4

F. Implications for Further Research 4

G. Special Comments 5

HIGH PERFORMANCE VIDEO COMPUTER

Quarterly Technical Report No. 1
Report Period: February 1, 1990 to October 15, 1990
Contract No. MDA 972-90-C-0022

A. TASK OBJECTIVES

The objective of this contract is to provide a continuous, real-time video system simulation capability to DARPA and the U.S. Government.

B. TECHNICAL PROBLEM

The simulation of video in real-time is a computationally intense problem. In the case of the NTSC video standard, a complete receiver function would require about 560 Giga-operations/s for real-time performance in which an operation is defined as a single bit operation. In the case of the newly proposed HDTV standards, the computational needs are enormous – growing to over 60 Tera-operations/s. In addition, the I/O system of such a computer must be able to sustain a data rate of at least 380 Mbytes/s.

C. GENERAL METHODOLOGY

The David Sarnoff Research Center (~~Sarnoff~~) has developed a massively parallel computer technology known as the Princeton Engine, which can meet the computational requirements noted above for the NTSC video standard. Special operational modes permit HDTV standards to be handled by vertically windowing the input and output HDTV images. In this way, the real-time instruction budget of the Princeton Engine is proportionally increased to handle the more demanding HDTV simulation problem.

Sarnoff has also developed a graphical programming environment for the Princeton Engine that goes beyond merely displaying video from a number of input streams in windows on the display. This environment assists the user in defining the processing of those video streams in the way a conventional workstation assists the programmer in the definition of software. Just as the concept of modular programming encourages the programmer to partition the overall problem into smaller, more tractable units, experience has shown that the block diagram paradigm is a particularly useful tool to characterize video processing. In this model, processing steps are represented by graphical entities connected into a network. The connecting lines represent the flow of video through the complete process while the details of the process are contained within each block. To achieve this end, the Princeton Engine is supported by a hierarchy of graphical programming tools.

The Princeton Engine graphical programming tools involve the use of modules from a library that are linked together to form a block diagram that serves as both the source code for the Princeton Engine and the documentation of the simulation idea. The resulting block diagram is then compiled into microcode, loaded into the engine, and executed in real-time. The user may alter parameters on the block diagram during run-time to observe, instantaneously, the results of the modifications. This could, for example, include filter coefficients, thresholds, delays, and algorithm control switches. Further, to assist in debugging, special software "probes" are provided to allow users to look at intermediate data throughout the block diagram during the real-time simulation.

Under this contract, Sarnoff will build and deliver to the National Institute of Standards and Technology (NIST) a turn-key Princeton Engine that is uniquely capable of real-time video system simulations. Sarncff will include the graphical programming environment developed specifically for video and image system simulation.

To support users other than video and image processing engineers, as well as those video signal processing engineers not accustomed to graphical programming, Sarnoff will provide a Fortran Compiler as part of the total simulation system. This compiler will be realized by modifying an existing Fortran compiler already in place for another massively parallel computer.

In order to handle future HDTV standards in a continuous, full-frame-size manner, Sarnoff will also design a second generation of the Princeton Engine that is capable of a 100 Tera-operations/s computational rate. The input and output bandwidths of this machine will be in excess of 8 Gbits/s.

D. TECHNICAL RESULTS

During this reporting period significant progress was made towards building a 1024 processor Princeton Engine to be delivered to NIST. The Fortran Compiler work also progressed well during this period.

1. 1024 Processor Princeton Engine for NIST

All long lead parts were ordered and most were received. Early in the project, it was found that the cylinder (processor) board vendor, Multiwire, was going out of business. This required a redesign of the board that has been successfully completed. The new boards are more reliable, more manufacturable, and hold promise of improved performance (20% higher clock frequency). However, this redesign was accomplished with considerable unexpected work.

The local memory was also redesigned for increased capacity and flexibility. This work was done as part of a parallel effort for another client and is leveraged here to deliver a four -old increase in memory capacity with no contract cost increase. This capability will allow users to run HDTV algorithms that exceed the current real-time instruction budget on full-sized images. This redesign has been 100% successful and production runs are now in process.

Finally, the microsequencer was also redesigned to accommodate the Fortran Compiler. The new design handles loops, jumps, and goto's, has a much larger program memory, and a trace capability. Design has been completed and assembly is currently underway.

At this time, it is expected that delivery of the Princeton Engine to NIST will be possible in late February 1991 as scheduled. This delivery will include the Princeton Engine Graphical Programming Environment. The Fortran Compiler will be delivered later as described below.

2. Fortran Compiler for the Princeton Engine

We are working with COMPASS to provide a Fortran Compiler that uses as much of COMPASS's Fortran front-end as possible. Version 1.0 of the Princeton Engine Fortran Compiler will support many of the scalar and array features of Fortran 90. Examples of the array features are:

- Subscript triplets
- WHERE construct
- ALL, ANY, COUNT, and other array reduction intrinsic functions
- MERGE, SPREAD, and other array construction functions
- CSHIFT, EOSHIFT, and other array manipulation functions
- MAXLOC, MINLOC
- Vector and matrix multiply functions

A compiler requirements document was generated and issued on April 18, 1990. This document was used to direct an analysis effort, performed by Sarnoff and COMPASS, to specify the interfaces between COMPASS and Sarnoff software. This analysis effort has been 90% completed and delineates Princeton Engine hardware changes and an intermediate code, named "8-code," which a Sarnoff translator will turn into Princeton Engine source code. An implementation effort has just begun on this translator, as well as an interpreter, a linker, and a core function library. An analysis report is expected to be completed by the end of October 1990.

As noted above, implementation is just beginning. A spike compiler is scheduled for January 1991. The alpha and beta versions are expected to be delivered in June and September 1991, respectively.

3. Sarnoff Engine Design

Most of this work is scheduled to be done between March 1, 1991, and September 31, 1991. However, a low-level effort focussed on interprocessor communication options to find an effective topology for the interconnection network and routing protocol has begun. Statistics from a network simulator show that hierarchical topologies are especially promising. Curren investigations are directed towards multistaged networks in which each stage spans different physical entities, such as across ICs, across boards, or across cabinets.

E. IMPORTANT FINDINGS AND CONCLUSIONS

Interprocessor communication networks based a hierarchical topology that is determined by physical constraints will minimize message and data latency in a massively parallel architecture as well as yield lower hardware complexity.

F. IMPLICATIONS FOR FURTHER RESEARCH

Future research on interprocessor communications will attempt to further decrease routing time and hardware complexity by evaluating existing adaptive routing protocols and developing new ones tailored for hierarchical networks. Such routing protocols will take advantage of the large number of redundant paths at each layer of the hierarchical network by changing routing paths when network traffic congestion is encountered. Simulations have already shown that routing time can be improved by up to four times simply by changing the routing protocol and flow control mechanism. In addition, considerable effort will be made to avoid transmitting the message's header that specifies its destination address. The header may be fully eliminated for routing patterns known before run time, or partially eliminated by using routing cycles to schedule messages according to their destinations, thus rendering a portion of their header unnecessary. Ultimately, a routing protocol suiting the hierarchical topology will allow efficient interprocessor communication with minimal hardware complexity.

G. SPECIAL COMMENTS

The local memory of the Princeton Engine was redesigned under a parallel effort for a commercial client to provide a capability to run algorithms that exceed the current real-time instruction budget on full-sized HDTV images. This new memory capability, called videoclip, will be delivered with the NIST Princeton Engine and is described below.

Videoclip Operational Mode

The Princeton Engine has an instruction budget for real-time signal processing (910 instructions for NTSC signals). However, some complex algorithms may require n times the real-time instruction budget to execute completely. When n is between 1 and 4, real-time processing can be maintained by processing 1/n of the vertical dimension of the input signal. The horizontal dimension is not affected. In the case of very large n, as is the case for HDTV signal processing, videoclip is used.

Videoclip is a near-real-time simulation mode of the Princeton Engine. Instead of continuously capturing, processing, and displaying real-time signals, videoclip separates these three steps into three disjoint operations. A normal sequence of operation will be to capture the desired material in real-time, process this material in near real-time, and display the results in real-time. The local memory of the Princeton Engine processors has been increased to allow a large sequence of video frames to be captured and displayed in real-time. Up to eight seconds of one input signal and three output signals can be stored for use in algorithms such as HDTV frame rate conversion motion flow field analysis.